



实验报告

学院：计算机科学学院专业：网络空间安全 2025 年 4 月 7 日

姓名	高梦珊	学号	2024040731222	
班级	一班	指导老师	孔德华老师	
课程名称	C 语言程序设计实践		成绩	
实验名称	数组-1			

1. 实验目的

- 理解数组中元素的存储机制；
- 掌握数据的定义方法；
- 掌握对数组中元素的赋值和设置方法；

2. 实验内容

- 将一个长度为 8 的整型数组中的值按逆序存放；（数组中数据自己从键盘动态输入）
- 输入 8 个数据，然后按照由小到大的顺序输出；
- 从键盘输入一个 4*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。
- 编写一个程序，计算出给定矩阵 $a[3][3]$ 中主对角线元素的和。
- 打印出杨辉三角的前 12 行数据，格式为下三角样式；
- 输入一个 4*3 的矩阵，对其转置后输出；
- 编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

3. 实验环境

- ① windows 10
- ② Vc++6.0/Dev-C++

4. 实验方法和步骤（含设计）

1、将一个长度为 8 的整型数组中的值按逆序存放

```
#include <stdio.h>
#define SIZE 8
int main() {
    int data[SIZE];
    int pos,k;
    printf("请逐个输入元素: \n");
    for(k=SIZE-1; k>=0; k--)
    {
        scanf("%d", &data[k]);
    }
    pos=0;
    while (pos < SIZE) {
        printf("%4d", data[pos]);
        pos++;
    }
    return 0;
}
```

2、输入 8 个数据，然后按照由小到大的顺序输出；

```
#include <stdio.h>
int main()
{
    int i,j;
    float a[8],Transfer;
    printf("请逐个输入数据并用回车隔开: \n");
    for(i=0;i<8;i++)
    {
        scanf("%f",&a[i]);
    }
    for(i=1;i<=8;i++)
    {
        for(j=0;j<8-i;j++)
        {
            if(a[j]>a[j+1])
            {
                Transfer=a[j];
                a[j]=a[j+1];
                a[j+1]=Transfer;
            }
        }
    }
    printf("这八个数从小到大的排列为 :\n");
    for(i=0;i<8;i++)
    printf("%f\t",a[i]);
    return 0;
}
```

3、从键盘输入一个 4*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。

```
#include<stdio.h>
int main()
{
    int i,j,k=1,min,min_1,min_2,a[4][3];
    printf("请逐个输入数据: \n");
    for(i=0;i<4;i++)
    {
        for(j=0;j<3;j++)
        {
            scanf("%d",&a[i][j]);
        }
    }
    min=a[0][0];
    min_1=0;
    min_2=0;
    for(i=0;i<4;i++)
    {
        for(j=0;j<3;j++)
        {
            if(min>=a[i][j])
            {
                min=a[i][j];
                min_1=i;
                min_2=j;
            }
        }
    }
    printf("该矩阵中的最小值为: %d, 它在第 %d 行, 第 %d 列",min,min_1+1,min_2+1);
    return 0;
}
```

4、计算给出定矩阵 a[3][3] 中主对角线元素的和。

```
#include<stdio.h>
int main()
{
    int i,j;
    float sum=0,a[3][3];
    printf("请输入数据: \n");
    for(i=0;i<3;i++)
        for(j=0;j<3;j++)
            scanf("%f",&a[i][j]);
    for(i=0;i<3;i++)
        sum+=a[i][i];
    printf("该矩阵主对角元素之和为: %f",sum);
    return 0;
}
```

5、打印出杨辉三角的前 12 行数据，格式为下三角样式；

```
#include<stdio.h>
int main()
{
    int i,j;
    int a[12][12];
    for(i=0;i<12;i++)
        a[i][0]=a[i][i]=1;
    for(i=0;i<11;i++)
    {
        for(j=1;j<=i;j++)
            a[i+1][j]=a[i][j-1]+a[i][j];
    }
    for(i=0;i<12;i++)
    {
        for(j=0;j<=i;j++)
            printf("%d\t",a[i][j]);
        printf("\n");
    }
    return 0;
}
```

6、输入一个 4*3 的矩阵，对其转置后输出；

```
#include<stdio.h>
int main()
{
    int i,j,k=1;
    int a[4][3],b[3][4];
    printf("请输入数据: \n");
    for(i=0;i<4;i++)
        for(j=0;j<3;j++)
            scanf("%d",&a[i][j]);
    for(i=0;i<4;i++)
        for(j=0;j<3;j++)
            b[j][i]=a[i][j];
    printf("该矩阵的转置为: \n");
    for(i=0;i<3;i++)
        for(j=0;j<4;j++)
    {
        printf("%d,",b[i][j]);
        if(k++%4==0)
            printf("\n");
    }
    return 0;
}
```

7、编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

```
#include<stdio.h>
int main()
{
    int i,t,n,m=0,key=0;
    int a[11];
    printf("请输入十个有序元素: \n");
    for(i=0;i<10;i++)
        scanf("%d",&a[i]);
    printf("数组为: \n");
    for(i=0;i<10;i++)
        printf("%d\t",a[i]);
    printf("\n请输入待插入的元素: \n");
    scanf("%d",&n);
    for(i=9;i>=0;i--)
    {
        if(n<a[i])
            a[i+1]=a[i];
        else
        {
            a[i+1]=n;
            break;
        }
    }
    printf("新的有序数组为: \n");
    for(i=0;i<11;i++)
        printf("%d\t",a[i]);
    return 0;
}
```

5. 程序及测试结果

1、将一个长度为 8 的整型数组中的值按逆序存放

请逐个输入元素：

```
1  
3  
2  
4  
56  
7  
8  
9  
      9   8   7   56   4   2   3   1
```

```
Process exited after 7.037 seconds with return value 0  
请按任意键继续. . .
```

2、输入 8 个数据，然后按照由小到大的顺序输出；

请逐个输入数据并用回车隔开：

```
1  
4  
5  
2  
3  
6  
7  
89
```

这八个数从小到大的排列为：

```
1.000000    2.000000    3.000000    4.000000  
5.000000    6.000000    7.000000    89.000000
```

```
Process exited after 4.739 seconds with return value 0  
请按任意键继续. . .
```

3、从键盘输入一个 4*3 整型数组赋值，找出其中的最小值，并将该值和其行号与列号输出出来。

请逐个输入数据：

```
1  
4  
2  
5  
3  
5  
6  
9  
87  
4  
5  
2
```

该矩阵中的最小值为：1，它在第 1 行，第 1 列

```
Process exited after 6.925 seconds with return value 0  
请按任意键继续. . .
```

4、计算出给定矩阵 a[3][3] 中主对角线元素的和。

请输入数据：

```
1  
2  
5  
4  
7  
6  
3  
9  
8
```

该矩阵主对角元素之和为： 16.000000

```
-----  
Process exited after 5.934 seconds with return value 0  
请按任意键继续. . .
```

5、打印出杨辉三角的前 12 行数据，格式为下三角样式；

```
-----  
1  
1 1  
1 2 1  
1 3 3 1  
1 4 6 4 1  
1 5 10 10 5 1  
1 6 15 20 15 6 1  
1 7 21 35 35 21 7 1  
1 8 28 56 70 56 28 8 1  
1 9 36 84 126 126 84 36 9 1  
1 10 45 120 210 252 210 120 45 10 1  
1 11 55 165 330 462 462 330 165 55 11 1  
-----  
Process exited after 0.617 seconds with return value 0  
请按任意键继续. . .
```

6、输入一个 4*3 的矩阵，对其转置后输出；

请输入数据：

```
1  
2  
5  
4  
3  
5  
8  
7  
9  
66  
3  
5
```

该矩阵的转置为：

```
1,4,8,66,  
2,3,7,3,  
5,5,9,5,
```

```
-----  
Process exited after 7.397 seconds with return value 0  
请按任意键继续. . .
```

7、编写一个程序，把一个数插入到一个有序的有 10 个元素的数组中，并使插入后的数组仍为有序数组。

请输入十个有序元素：

1
2
3
4
5
6
7
8
9
12

数组为：

1 2 3 4 5 6 7 8 9 12

请输入待插入的元素：

10

新的有序数组为：

1 2 3 4 5 6 7 8 9 10 12

Process exited after 13.84 seconds with return value 0

请按任意键继续. . .

6. 实验分析与体会

本次实验让我在数组的操作与应用中获得了深刻的实践认知。从一维数组的动态输入到二维矩阵的复杂处理，每一步操作都加深了我对数据结构底层逻辑的理解。例如，在实现数组逆序时，我最初选择倒序输入直接完成存储，虽然结果正确，但后来意识到“首尾交换”更能体现数据操作的完整性，这让我反思代码设计应兼顾功能实现与逻辑清晰性。通过多次调试，我逐渐掌握了如何通过循环精准控制数组元素的赋值与修改，尤其是在处理用户输入时，明确格式提示（如“输入格式：a, b, c”）能有效避免数据错位，这一经验让我对程序交互设计有了新的认识。

排序算法的实践让我体会到基础逻辑的重要性。在冒泡排序中，通过相邻元素的反复比较与交换，我直观感受到算法效率的差异。尽管冒泡排序在处理小规模数据时表现尚可，但其时间复杂度让我意识到未来学习中需要掌握更高效的排序策略。此外，在二维数组极值查找中，双重循环的嵌套使用让我熟悉了行列索引的配合，而初始化最小值为首先元素后，通过严格比较条件（如“小于而非小于等于”）确保记录到首个最小值，这一细节让我深刻体会到初始条件设置对结果的影响。

矩阵相关任务进一步扩展了我的编程视野。主对角线求和的实现让我学会利用索引规律($i=j$)快速定位元素，而矩阵转置则通过行列互换揭示了数据重组的本质。这些操作让我看到数组在数学计算中的实际价值，例如转置在矩阵运算中的基础性作用。在有序数组插入实验中，从后向前遍历并逐个后移元素的过程，既巩固了数组连续存储的特性，也让我意识到静态数组的局限性——频繁插入场景下需结合动态内存分配提升灵活性。

调试过程中的教训同样珍贵。例如，浮点数输出未限制小数位导致显示混乱，让我学会用格式控制符规范结果；冒泡排序外层循环边界错误引发的越界风险，则强化了我对循环条件严谨性的重视。这些经历让我明白，编程不仅是逻辑构建，更是对细节的极致把控。通过本次实验，我不仅掌握了数组的核心操作，更在实践中培养了系统化的编程思维，为后续探索更复杂的数据结构积累了宝贵经验。

实验日期 : 2025 年 4 月 7 日

教师评语

签名: _____ 年 ____ 月 ____ 日